

# **GR551x AT Command Example Application**

Version: 1.3

Release Date: 2022-02-20

Shenzhen Goodix Technology Co., Ltd.

### Copyright © 2022 Shenzhen Goodix Technology Co., Ltd. All rights reserved.

Any excerption, backup, modification, translation, transmission or commercial use of this document or any portion of this document, in any form or by any means, without the prior written consent of Shenzhen Goodix Technology Co., Ltd. is prohibited.

### **Trademarks and Permissions**

**GODIX** and other Goodix trademarks are trademarks of Shenzhen Goodix Technology Co., Ltd. All other trademarks and trade names mentioned in this document are the property of their respective holders.

### Disclaimer

Information contained in this document is intended for your convenience only and is subject to change without prior notice. It is your responsibility to ensure its application complies with technical specifications.

Shenzhen Goodix Technology Co., Ltd. (hereafter referred to as "Goodix") makes no representation or guarantee for this information, express or implied, oral or written, statutory or otherwise, including but not limited to representation or guarantee for its application, quality, performance, merchantability or fitness for a particular purpose. Goodix shall assume no responsibility for this information and relevant consequences arising out of the use of such information.

Without written consent of Goodix, it is prohibited to use Goodix products as critical components in any life support system. Under the protection of Goodix intellectual property rights, no license may be transferred implicitly or by any other means.

### Shenzhen Goodix Technology Co., Ltd.

Headquarters: 2F. & 13F., Tower B, Tengfei Industrial Building, Futian Free Trade Zone, Shenzhen, China

TEL: +86-755-33338828 FAX: +86-755-33338099

Website: www.goodix.com

# Preface

#### Purpose

This document introduces how to use and verify the ble\_app\_uart\_at example in the GR551x Software Development Kit (SDK), to help users quickly get started with secondary development.

### Audience

This document is intended for:

- GR551x user
- GR551x developer
- GR551x tester
- Hobbyist developer

### **Release Notes**

This document is the fourth release of *GR551x AT Command Example Application*, corresponding to GR551x Systemon-Chip (SoC) series.

#### **Revision History**

| Version | Date       | Description  |
|---------|------------|--|
| 1.0     | 2021-02-23 | Initial release  |
| 1.1     | 2021-04-20 | Optimized descriptions in "Initial Operation" and "Application Details". |
| 1.2     | 2021-08-09 | Changed the section "Supported Development Platform" into "Preparation". |
| 1.3     | 2022-02-20 | Modified the file name of the example firmware based on SDK changes.     |

# Contents

| I    |
|------|
| 1    |
| 2    |
| 3    |
| 3    |
| 3    |
| 4    |
| 8    |
| 8    |
| . 10 |
| 10   |
| . 11 |
| . 12 |
| .13  |
| . 14 |
| 16   |
| 18   |
| .18  |
| .18  |
| . 18 |
| . 19 |
| .19  |
| 21   |
|      |

# GODIX

# **1** Introduction

GR551x Software Development Kit (SDK) provides an AT-command-related example, ble\_app\_uart\_at, to help developers quickly build a Bluetooth module and enable Bluetooth Low Energy (Bluetooth LE) communications. The example allows developers to control hardware through simple AT commands based on actual demands, freeing them up from modifying source code. AT commands feature easy extension and can be easily customized by users based on actual demands.

AT commands can be used to start/stop advertising, set advertising parameters, start/stop scanning, set scanning parameters, and get device name/address. In addition, the commands support control of devices equipped with SoCs through terminals. This makes integration of ble\_app\_uart\_at into third-party microcontrollers possible.

This document introduces how to use and verify the ble\_app\_uart\_at example in the GR551x SDK.

Before getting started, you can refer to the following documents.

| Name                      | Description  |
|---------------------------|--|
| GR551x Developer Guide    | Introduces GR551x SDK and how to develop and debug applications based on the SDK.  |
| J-Link/J-Trace User Guide | Provides J-Link operational instructions. Available at <u>www.segger.com/downloads/jlink/</u><br><u>UM08001_JLink.pdf</u> .                              |
| Keil User Guide           | Offers detailed Keil operational instructions. Available at <a href="http://www.keil.com/support/man/docs/uv4/">www.keil.com/support/man/docs/uv4/</a> . |

#### Table 1-1 Reference documents

# 2 Profile Overview

Based on Goodix UART Service (GUS), the ble\_app\_uart\_at example is mainly used to enable passthrough. As the most easy-to-use means of Bluetooth LE communications, passthrough features:

- Unaltered service data during transmission
- Bidirectional data transfer

The GUS is identified by its vendor-specific Universally Unique Identifier (UUID), A6ED0201-D344-460A-8075-B9E8EC90D71B.

GUS includes three characteristics:

- GUS TX characteristic: Transmits data.
- GUS RX characteristic: Receives data.
- GUS Flow Control characteristic: Controls data flow.

These characteristics are described in detail as follows:

#### Table 2-1 GUS characteristics

| Characteristic   | UUID                                 | Туре     | Support   | Security | Property     |
|------------------|--------------------------------------|----------|-----------|----------|--------------|
| GUS TX           | A6ED0203-D344-460A-8075-B9E8EC90D71B | 128 bits | Mandatory | None     | Notify       |
| GUS RX           | A6ED0202-D344-460A-8075-B9E8EC90D71B | 128 bits | Mandatory | None     | Write        |
| GUS Flow Control | A6ED0204-D344-460A-8075-B9E8EC90D71B | 128 bits | Mandatory | None     | Write/Notify |

# **3** Initial Operation

This chapter introduces how to run and verify the ble\_app\_uart\_at example in the GR551x SDK.

### Note:

SDK\_Folder is the root directory of GR551x SDK.

### **3.1 Preparation**

Perform the following tasks before running the ble\_app\_uart\_at example.

• Hardware preparation

Table 3-1 Hardware preparation

| Name              | Description   |
|-------------------|---|
| Development board | GR5515 Starter Kit Board (SK Board); two boards are required. |
| Connection cable  | Micro USB 2.0 cable   |

#### • Software preparation

Table 3-2 Software preparation

| Name                  | Description  |
|-----------------------|--|
| Windows               | Windows 7/Windows 10   |
| J-Link driver         | A J-Link driver. Available at <u>www.segger.com/downloads/jlink/</u> .   |
| Keil MDK5             | An integrated development environment (IDE). MDK-ARM Version 5.20 or later is required. Available at <u>www.keil.com/download/product/</u> . |
| GProgrammer (Windows) | A programming tool. Available in SDK_Folder\tools\GProgrammer.   |
| GRUart (Windows)      | A serial port debugging tool. Available in SDK_Folder\tools\GRUart.  |

### 3.2 Firmware Programming

The source code of the ble\_app\_uart\_at example is in SDK\_Folder\projects\ble\ble\_multi\_role\ble\_ app\_uart\_at.

You can programme *ble\_app\_uart\_at.bin* to an SK Board through GProgrammer. For details, see *GProgrammer User Manual*.

For a project involving modification on source code of ble\_app\_uart\_at, re-compile the project to generate a new *ble\_app\_uart\_at.bin* file, and then programme the file to the SK Board. For details, refer to *GR551x Developer Guide*.

### Dote:

• The *ble\_app\_uart\_at.bin* is in SDK\_Folder\projects\ble\_multi\_role\ble\_uart\_at\buil d.

# G@DiX

# 3.3 Test and Verification

In this test, two SK Boards are required, with one named as SK Board A (as Client) and the other as SK Board B (as Server). The two boards communicate with each other through Bluetooth LE.

After the SK Boards and GRUart are ready, start GRUart. Wait until GRUart displays device address information and **Goodix UART(AT) example start**. This indicates the ble\_app\_uart\_at firmware operates properly. The figure below shows the proper operation of the firmware on SK Board B.

| 🥌 GRUart                        |                                   |                           |                                  |           |        | - C     | ) ×    |
|---------------------------------|-----------------------------------|---------------------------|----------------------------------|-----------|--------|---------|--------|
| Serial Port Setting             | - <b>Receive Dat</b> :<br>Format: | a 💿 ASCII                 | ⊖ Hex                            | Show      | Time 🗌 | Font Si | ze 10  |
| PortName COM60 JLink CDC $\sim$ | Background:                       | 🖲 White                   | O Black                          |           |        |         | Search |
| BaudRate 115200 V               | APP_I: Local 1<br>APP_I: Goodix   | Board EA:CH<br>UART(AT) ( | 3:3E:45:23:01.<br>example start. |           |        |         |        |
| DataBits 8 🗸                    |                                   |                           |                                  |           |        |         |        |
| Parity None 🗸                   |                                   |                           |                                  |           |        |         |        |
| StopBits 1                      |                                   |                           |                                  |           |        |         |        |
| Flow Control 🗌 RTS 🗌 DTR        |                                   |                           |                                  |           |        |         |        |
| Close Port                      |                                   |                           |                                  |           |        |         |        |
|                                 | 1                                 |                           |                                  |           | Save   | Pause   | Clear  |
| -Tx <u>R</u> x Data Size        | Send data                         |                           |                                  |           |        |         |        |
|                                 | Single Multi                      |                           |                                  |           |        |         |        |
| Tx Count 0 Bytes                | Format: 🖲 AS                      | CII () Hex                | Loop 🗌                           | Period 50 | т.     | us 🗌 Ne | wLine  |
| Rx Count 78 Bytes               |                                   |                           |                                  |           |        |         |        |
| Clear                           |                                   |                           |                                  |           |        |         |        |
|                                 | file path                         |                           |                                  | Browse    | Send   | Pause   | Clear  |
| Port Opened CTS=1 DSR=1 DCD=0   | )                                 |                           |                                  |           |        |         |        |

Figure 3-1 GRUart information displaying proper operation of firmware

#### **Note**:

The device address displayed on GRUart is the one generated after modifying source code of the ble\_app\_uart\_at example. The actual device address used by users prevails.

After the firmware of the ble\_app\_uart\_at example operates normally, deliver AT commands to perform specific Bluetooth LE operations.

1. Send AT: ADV\_STOP command to SK Board B to stop advertising, after which send AT: ADV\_START command to restart advertising.



| GRUart                       |  | - 0                  | ×      | GRUart                     |                                       |  |           | _        |          |
|------------------------------|--|----------------------|--------|----------------------------|---------------------------------------|--|-----------|----------|----------|
|                              | Receive Data   |                      |        |                            | Receive Data                          |  |           |          |          |
| Serial Port Setting          | Format:      ASCII O Hex   | Show Time 🗌 Font Sis | ze 10  | Serial Port Setting        | Format:                               | ● ASCII ○ Hex                                      | Show Tim  | e 🗌 Font | Size 10  |
| PortName CON60 JLink CDC ~   | Background:   White   Black  | 2                    | Search | PortName COM48 JLink CDC   | Background:                           | • White 🔘 Black                                    |           |          | Search   |
| BaudRate 115200 ~            | APP_I: Local Board EA:CB:3E:45:23:01.<br>APP_I: Goodix UART(AT) example start. |                      |        | BaudRate 115200            | APP_I: Local B<br>APP_I: Goodix<br>OK | oard EA:CB:3E:45:23:01.<br>UART(AT) example start. |           |          |          |
| DataBits 8 🗸                 |  |                      |        | DataBits 8                 | OK                                    |  |           |          |          |
| Parity None ~                |  |                      |        | Parity None                | ~                                     |  |           |          |          |
| StopBits 1                   |  |                      |        | StopBits 1                 | ~                                     |  |           |          |          |
| Flow Control _ RIS _ DIR     |  |                      |        | Flow Control RTS D         | ſR                                    |  |           |          |          |
| Close Port                   |  |                      |        | Close Port                 |                                       |  |           |          |          |
|                              |  | Save Pause           | Clear  |                            | ,                                     |  | S         | ave Pau: | se Clear |
| Tx <u>R</u> x Data Size      | Send data<br>Single Multi  |                      |        | TxRx Data Size             | Send data<br>Single Multi             |  |           |          |          |
| Tx Count 13 Bytes            | Format:      ASCII      Hex Loop      Ho                                       | eriod 50 🔹 ms 🗹 New  | Line   | Tx Count 36 Byte           | es Format:      ASC                   | II 🔿 Hex 🛛 Loop 🗌                                  | Period 50 | 🕈 ms 🗹   | NewLine  |
| Rx Count 82 Bytes            | AT:ADV_STOP  |                      |        | Rx Count 90 Byte           | AT:ADV_START                          |  |           |          |          |
| Clear                        |  |                      |        | Clear                      |                                       |  |           |          |          |
|                              | file path  | Browse Send Pause    | Clear  |                            | file path                             |  | Browse Se | nd Paus  | e Clear  |
| Port Opened CTS=1 DSR=1 DCD= | 0  |                      |        | Port Opened CTS=1 DSR=1 DC | CD=0                                  |  |           |          |          |

Figure 3-2 Stopping advertising

Figure 3-3 Restarting advertising

2. Send AT: SCAN\_START command to SK Board A to start scanning. When SK Board A discovers the GUS, send A T: CONN\_INIT= command to initiate a connection with SK Board B.

| 🙆 GRUart                      |  | – 🗆 X                             | GRUart                       |  | – 🗆 ×                    |
|-------------------------------|--|-----------------------------------|------------------------------|--|--------------------------|
| Sanial Bank Sataira           | Receive Data   |                                   | Coviel Dart Cotting          | Receive Data   | Share Trian Day 1 and 10 |
| Serial Fort Setting           | Format:      ASCII O Hex   | Show line Font Size 10            | Serial Fort Setting          | Format: I ASCII O Hex  | Show line Pont Size 10   |
| PortName COM48 JLink CDC ~    | Background:      White  Black  | Search                            | PortName COM48 JLink CDC ~   | Background:  White O Black   | Search                   |
| BaudRate 115200               | APP_I: Local Board EA:CB:3E:22:11:00.<br>APP_I: Goodix UART(AT) example start.<br>OK |                                   | BaudRate 115200 ~            | APP_I: Local Board EA:CB:3E:22:11:00.<br>APP_I: Goodix UART(AT) example start.<br>OK |                          |
| DataBits 8                    | Target Device Found  |                                   | DataBits 8 ~                 | Target Device Found<br>CONNECTED<br>APP I: Goodin Hart Service discovery con         | alatalır                 |
| Parity None 🗸                 |  |                                   | Parity None ~                | APP_I: Enabled TX Notification.<br>APP_I: Enabled Flow Control Notification.         |                          |
| StopBits 1                    |  |                                   | StopBits 1                   |  |                          |
| Flow Control RTS DTR          |  |                                   | Flow ControlRTSDTR           |  |                          |
| Close Port                    |  |                                   | Close Port                   |  |                          |
|                               |  | Save Pause Clear                  |                              | 1  | Save Pause Clear         |
| Tx <u>R</u> x Data Size       | Send data  |                                   | Tx <u>R</u> x Data Size      | Send data  |                          |
|                               | Single Multi   |                                   |                              | Single Multi   |                          |
| Tx Count 15 Bytes             | Format:      ASCII      Hex Loop      Pe   | riod <sup>50</sup> 🛊 ms 🗹 NewLine | Tx Count 30 Bytes            | Format:  ASCII O Hex Loop Per  | riod 50 🔹 ns 🗹 NewLine   |
| Rx Count 279 Bytes            | AT:SCAN_START  |                                   | Rx Count 416 Bytes           | AT:CONN_INIT=  |                          |
| Clear                         |  |                                   | Clear                        |  |                          |
|                               | file path B  | rowse Send Pause Clear            |                              | file path Br   | rowse Send Pause Clear   |
| Port Opened CTS=1 DSR=1 DCD=0 |  |                                   | Port Opened CTS=1 DSR=1 DCD= | 0  |                          |

Figure 3-4 Starting scanning

Figure 3-5 Initiating a connection after discovering GUS

- 3. After SK Board A is successfully connected to SK Board B, execute specific AT commands to get the address and role information of the two boards.
  - Send AT: ADDR? command to get address information.



– 🗆 🗙

| 🙆 GRUart                        |  |                     | 1 × 1  | GRUart                          |  |   |             |        | - C     | ) >    |
|---------------------------------|--|---------------------|--------|---------------------------------|--|---|-------------|--------|---------|--------|
| Serial Port Setting             | Receive Data<br>Format:      ASCII O Hex   | Show Time 🗌 Font Si | ze 10  | Serial Port Setting             | Receive Dat                                    | ● ASCII ○ Hex   | Show        | Time 🗌 | Font Si | ize 10 |
| PortName COM48 JLink CDC $\sim$ | Background:  White  Black  |                     | Search | PortName COM60 JLink CDC $\sim$ | Background:                                    | • White 🔘 Black   |             |        |         | Search |
| BaudRate 115200 ~               | APP_I: Local Board EA:CB:3E:22:11:00.<br>APP_I: Goodix UART(AT) example start.<br>OK |                     |        | BaudRate 115200 ~               | APP_I: Local<br>APP_I: Goodix<br>0-EA:CB:3E:45 | Board EA:CB:3E:45:23:<br>UART(AT) example sta<br>:23:01 | 01.<br>.rt. |        |         |        |
| DataBits 8 ~                    | Target Device Found<br>CONNECTED<br>APP I: Goodin Hart Service discovery on          |                     |        | DataBits 8 ~                    |  |   |             |        |         |        |
| Parity None ~                   | APP_I: Enabled TX Notification.<br>APP_I: Enabled Flow Control Notificatio:          | n.                  |        | Parity None ~                   |  |   |             |        |         |        |
| StopBits 1 ~                    | 0-EA:CB:3E:22:11:00  |                     |        | StopBits 1 ~                    |  |   |             |        |         |        |
| Flow Control 🗌 RIS 🔹 DIR        |  |                     |        | Flow Control _ RTS _ DTR        |  |   |             |        |         |        |
| Close Port                      |  |                     |        | Close Port                      |  |   |             |        |         |        |
|                                 |  | Save Pause          | Clear  |                                 | 1  |   |             | Save   | Pause   | Clea   |
| Tx <u>R</u> x Data Size         | Send data<br>Single Multi  |                     |        | Tx <u>R</u> x Data Size         | Send data<br>Single Multi                      |   |             |        |         |        |
| Tx Count 49 Bytes               | Format:      ASCII      Hex Loop      P  | eriod 50 🗦 ns 🗹 New | wLine  | Tx Count 10 Bytes               | Format: 🖲 AS                                   | SCII 🔾 Hex Loop   | Period 5    | , 🗘 0  | ns 🗹 Ne | wLine  |
| Rx Count 463 Bytes              | AT:ADDR?   |                     |        | Rx Count 255 Bytes              | AT:ADDR?                                       |   |             |        |         |        |
| Clear                           |  |                     |        | Clear                           |  |   |             |        |         |        |
|                                 | file path  | Browse Send Pause   | Clear  |                                 | file path                                      |   | Browse      | Send   | Pause   | Clear  |
| Port Opened CTS=1 DSR=1 DCD=    | 0  |                     |        | Port Opened CTS=1 DSR=1 DCD=    | :0   |   |             |        |         |        |

🥌 GRUart

Figure 3-6 Getting device address information of SK Board A

| Figure 3-7 | Getting device | address | information | of SK | Board | В |
|------------|----------------|---------|-------------|-------|-------|---|
|------------|----------------|---------|-------------|-------|-------|---|

Send AT: GAP\_ROLE? command to get role information. •

| 🙆 GRUart          |               |         |   |  |               |           |        | -    |       | ×     |
|-------------------|---------------|---------|---|--|---------------|-----------|--------|------|-------|-------|
| Serial P          | ort Setting   |         | -Receive Dat:<br>Format:  | <ul> <li>ASCII</li> <li>ND:11</li> </ul> | O Hex         | Show      | Time 🗌 | Font | Size  | 10    |
| PortName          | COH48 JLink C | IDC 🗸   | Background:   | • White                                  | OBlack        |           |        |      | Sea   | irch  |
| BaudRate          | 115200        | $\sim$  | APP_1: Local J<br>APP_I: Goodix<br>OK   | Joard EA:CE<br>UART(AT) e                | xample start. |           |        |      |       |       |
| DataBits          | 8             | $\sim$  | Target Device<br>CONNECTED  | Found                                    |               |           |        |      |       |       |
| Parity            | None          | $\sim$  | APP_I: Goodix Uart Service discovery completely.<br>APP_I: Enabled TX Notification. |  |               |           |        |      |       |       |
| StopBits          | 1             | ~       | 0-EA: CB: 3E: 22<br>CENTRAL   | 11:00                                    |               |           |        |      |       |       |
| Flow Cont         | rol RIS       | DTR     |   |  |               |           |        |      |       |       |
|                   |               |         | 1   |  |               |           | Save   | Paus | se i  | Clear |
| Tx <u>R</u> x Dat | a Size        |         | Send data<br>Single Multi   |  |               |           |        |      |       |       |
| Tx Count          | 63            | Bytes   | Format: 🖲 ÅS  | CII 🔾 Hex                                | Loop 🗌        | Period 50 | ÷ 1    | is 🗹 | NevLi | .ne   |
| Rx Count          | 472           | Bytes   | AT:GAP_ROLE   | 3  |               |           |        |      |       |       |
|                   | Clear         |         |   |  |               |           |        |      |       |       |
|                   |               |         | file path   |  |               | Browse    | Send   | Paus | e C   | lear  |
| Port Openeo       | d CTS=1 DSR=  | 1 DCD=0 |   |  |               |           |        |      |       |       |

|          |  |   | 0   |  |  | Fonc 51  | ze 10  |
|----------|--|---|---|--|--|--|--|
| nk CDC 🗸 | Background:                                      | ● White   | ⊖ Black   |  |  |  | Search   |
| ~        | APP_I: Local I<br>APP_I: Goodix<br>0-EA:CB:3E:45 | Board EA:CE<br>UART(AT) e<br>:23:01   | :3E:45:23:01.<br>xample start.  |  |  |  |  |
| $\sim$   | PERIPHERAL                                       |   |   |  |  |  |  |
| $\sim$   |  |   |   |  |  |  |  |
| $\sim$   |  |   |   |  |  |  |  |
| DTR      |  |   |   |  |  |  |  |
| ct       |  |   |   |  |  |  |  |
|          | 1  |   |   |  | Save   | Pause  | Clear  |
|          | Send data<br>Single Multi                        |   |   |  |  |  |  |
| Bytes    | Format: 🖲 AS                                     | CII 🔾 Hex   | Loop 🗌  | Period 50  | <b>ф</b> л   | us 🗹 Ne  | wLine  |
| Bytes    | AT:GAP_ROLE                                      | ?   |   |  |  |  |  |
|          |  |   |   |  |  |  |  |
|          | file path  |   |   | Browse   | Send   | Pause  | Clear  |
|          | ct Bytes   | APP_I: Local J<br>OFA.CB.3E.45<br>PERIFHERAL<br>DTR<br>et<br>Send data<br>Single Multi<br>Fornat:@h25<br>Bytes<br>File path | APP_I: Local Board EA:C<br>APP_I: Cocal MERT(A):<br>O-EA:CB:3E:45:23:01<br>PERIPHERAL<br>DIR<br>Single Multi<br>Bytes<br>Bytes<br>File path | APP_I: Local Board EA.CB: 3E:46:23:01<br>PERFECTION AND AND AND AND AND AND AND AND AND AN | APP_I: Local Board EA.CB: 38: 45: 23: 01.<br>APP_I: Goods UNKT(AT) example start.<br>O-BA.CB: 38: 45: 23: 01<br>PERIPHERAL<br>DIR<br>Send data<br>Single Multi<br>Porma: @ ASCII O Hex Loop Period <sup>50</sup><br>File path Browse | APP_I: Coola Board EA CO: 38:46:23:01.<br>APP_I: Coola BOard EA CO: 38:46:23:01<br>PERTMERAL<br>O-EA CE 38:46:23:01<br>PERTMERAL<br>DTR<br>Save<br>Send data<br>Single Multi<br>Pormat: © ASCII O Hex Loop D Period 50 S<br>ATGAP_ROLE?<br>File path Browse Send | APP.I. Cooli Board EA. (5: 38: 46: 23: 01.<br>APP.I. Goolin URX (AT) example start.<br>O-EA. (B: 36: 45: 23: 01<br>PERTPIBERAL |

Figure 3-9 Getting role information of SK Board B

- 4. Enable data transmission via GUS.
  - SK Board B (Server) sends Goodix\_BLE to SK Board A (Client). •



| GRUart  | - 🗆 X           | GRUart  |   |   |           | _        |         | ×     |
|---|-----------------|---|---|---|-----------|----------|---------|-------|
| Serial Port Setting Receive Data<br>Fornat: @ ASCII O Hex Show Time   | Font Size 10    | rial Port Setting   | Receive Data<br>Format:                         | ASCII O Hex                                 | Show 3    | fime 🗌 F | ont Siz | e 10  |
| PortName COM60 JLink CDC - Background:  White O Black   | Search          | tName COM48 JLink CDC $\scriptstyle{\scriptstyle \smallsetminus}$ | Background:                                     | White O Black                               |           |          | S       | earch |
| BaudRate 115200 APP_1: Good Batta BAtCB: SE: 40: 23:01:<br>APP_1: Good Batta UART(AT) example start.<br>0-EA: CB: 3E: 45: 23:01 | Bau             | dRate 115200 ~  | APP_I: Local Bo<br>APP_I: Goodix U<br>OK        | MART(AT) example start.                     |           |          |         |       |
| DataBits 8 PERIPHERAL   | Dat             | aBits 8 ~   | Target Device F<br>CONNECTED<br>APP_I: Goodix U | found<br>Jart Service discovery c           | ompletely |          |         |       |
| Parity None   | Par             | ity None 🗸  | APP_I: Enabled<br>APP_I: Enabled                | TX Notification.<br>Flow Control Notificati | on.       |          |         |       |
| StopBits 1  | Sto             | pBits 1 ~   | 0-EA: CB: 3E: 22: 1<br>CENTRAL                  | 1:00  |           |          |         |       |
| Flow Control RIS DTR  | Flo             | ow Control 🗌 RIS 🗌 DIR  | o o o d i a j i i i i i i i i i i i i i i i i i |   |           |          |         |       |
| Close Port  |                 | <ul> <li>Close Port</li> </ul>                                    |   |   |           |          |         |       |
| St  | ave Pause Clear |   |   |   |           | Save     | Pause   | Clear |
| TxRx Data Size Send data<br>Single Multi  | TxI             | <u>R</u> x Data Size  | Send data<br>Single Multi                       |   |           |          |         |       |
| Tx Count 36 Bytes Format:  ASCII O Hex Loop Period 50   | ns NewLine Tx   | Count 63 Bytes  | Format:     ASC                                 | II 🔾 Hex Loop 🗌                             | Period 50 | 🕈 ns     | 🗹 Newl  | Line  |
| Rx Count 267 Bytes  | Rx              | Count 484 Bytes   |   |   |           |          |         |       |
| Clear<br>file path Browse Se  | nd Pause Clear  | Clear   | file path                                       |   | Browse    | Send I   | ause    | Clear |
| Port Opened CTS=1 DSR=1 DCD=0   | Port            | Opened CTS=1 DSR=1 DCD=0  |   |   |           |          |         |       |

Figure 3-10 Server sends data to Client

• SK Board A (Client) sends Hello Word! to SK Board B (Server).

| 🥘 GRUart   |                                       |  |                          |                                  |            |          | -      |       | ×     |
|------------|---------------------------------------|--|--------------------------|----------------------------------|------------|----------|--------|-------|-------|
| Serial F   | ort Setting                           | -Receive Data<br>Format:                   | • ASCII                  | ⊖ Hex                            | Show       | Time 🗌   | Font   | Size  | 10    |
| PortName   | COM48 JLink CDC ${\scriptstyle \lor}$ | Background:                                | ) White                  | ○ Black                          |            |          |        | Sea   | arch  |
| BaudRate   | 115200 ~                              | APP_I: Local B<br>APP_I: Goodix<br>OK      | oard EA:CH<br>UART(AT) e | 3:3E:22:11:00.<br>example start. |            |          |        |       |       |
| DataBits   | 8 ~                                   | CONNECTED                                  | Found                    |                                  |            |          |        |       |       |
| Parity     | None $\vee$                           | APP_I: Goodix<br>APP_I: Enabled            | TX Notifi                | ication.                         | completely | ·.       |        |       |       |
| StopBits   | 1 ~                                   | 0-EA: CB: 3E: 22:<br>CENTRAL<br>Condin BLE | 11:00                    | .ioi wotilitat.                  | 101.       |          |        |       |       |
| Flow Cont  | rol 🗌 RTS 🗌 DIR                       | SOUGHX_BLE                                 |                          |                                  |            |          |        |       |       |
| •          | Close Port                            |  |                          |                                  |            |          |        |       |       |
|            |                                       | 1  |                          |                                  |            | Save     | Paus   | e     | Clear |
| TxRx Dat   | a Size                                | Send data                                  |                          |                                  |            |          |        |       |       |
|            |                                       | Single Multi                               |                          |                                  |            |          |        |       |       |
| Tx Count   | 76 Bytes                              | Format:     ASC                            | II 🔿 Hex                 | Loop 🗌                           | Period 50  | <b>т</b> | us 🗹 I | NewLi | ne    |
| Rx Count   | 484 Bytes                             | Hello Word!                                |                          |                                  |            |          |        |       |       |
|            | Clear                                 |  |                          |                                  |            |          |        |       |       |
|            |                                       | file path                                  |                          |                                  | Browse     | Send     | Pause  | e (   | lear  |
| Port Opene | - CTS-1 DSP-1 DCD-0                   |  |                          |                                  |            |          |        |       |       |

Figure 3-12 Client sends data to Server

| 🔄 GRUart  |                 |                          |                           |                    |           |        | - 0      | I X    |
|-----------|-----------------|--------------------------|---------------------------|--------------------|-----------|--------|----------|--------|
|           |                 | Receive Dat              | 8                         |                    |           |        |          |        |
| Serial H  | ort Setting     | Fornat:                  | <ul> <li>ASCII</li> </ul> | ⊖ Hex              | Show      | Time 🗌 | Font Si  | ze 10  |
| PortName  | COM60 JLink CDC | Background:              | • White                   | ○ Black            |           |        |          | Search |
| DaudPata  |                 | APP_I: Local             | Board EA:CH               | 3: 3E: 45: 23: 01. |           |        |          |        |
| Dauditate | 115200          | 0-EA: CB: 3E: 45         | 23:01                     | example start.     |           |        |          |        |
| DataBits  | 8               | PERIPHERAL<br>Hello Word |                           |                    |           |        |          |        |
|           | -               | insite ford.             |                           |                    |           |        |          |        |
| Farity    | None            | ~                        |                           |                    |           |        |          |        |
| StopBits  | 1               | $\sim$                   |                           |                    |           |        |          |        |
| Flow Cont | Close Fort      | IR                       |                           |                    |           |        |          |        |
|           |                 |                          |                           |                    |           | Save   | Pause    | Clear  |
| 7 . P. D  |                 |                          |                           |                    |           |        |          |        |
| IXKX Dat  | a 51ze          | Single Multi             |                           |                    |           |        |          |        |
| Tx Count  | 36 Byt          | es Format: 🖲 AS          | CII 🔿 Hex                 | Loop 🗌             | Period 50 | ÷ ,    | ıs 🗹 Nev | Line   |
| Rx Count  | 280 Byt         | es                       |                           |                    |           |        |          |        |
|           | Clear           |                          |                           |                    |           |        |          |        |
|           |                 | file path                |                           |                    | Browse    | Send   | Pause    | Clear  |
|           | J CTC-1 DCR-1 D | -0                       |                           |                    |           |        |          |        |

Figure 3-13 Server receives data from Client

# **4** Application Details

This chapter introduces the running procedures and major code of the ble\_app\_uart\_at example.

### **4.1 Running Procedures**

This section elaborates on the running procedures of the ble\_app\_uart\_at example, aiming to help users deeply understand the operational mechanism of the example.

The following figure displays the running procedures of the ble\_app\_uart\_at example:



Figure 4-1 ble\_app\_uart\_at running procedures

- 1. Judge whether the received data is an AT command. If yes, check and update the environment variables of the AT command. If no, perform Step 4.
- 2. Read the environment variables of the AT command. When there is an AT command to be executed and its command handler is not null, perform Step 3.

- 3. Perform Bluetooth LE operations corresponding to the AT command.
- 4. Write to-be-transmitted data to ring buffers. The ring buffers comprise two types: ble to uart buffer (for storing received data) and uart to ble buffer (for storing to-be-transmitted data).
  - When AT commands are sent via GRUart, the command execution result is cached to the ble to uart buffer.
  - When non-AT commands are sent via GRUart, the data transmission mechanism is explained by taking two SK Boards running ble\_app\_uart\_at firmware as an example. Connect the two SK Boards to a PC and enable Bluetooth on the boards. SK Board A (Client) sends non-AT commands to the SK Board B (Server) via GRUart. The to-be-transmitted data will be cached into the uart to ble buffer before transmission by the Client, and the Server caches the received data to the ble to uart buffer.
- 5. Read the ring buffers to check whether there is data to be transmitted. If yes, transmit the data. Otherwise, return to Step 1.

The following figure displays how to execute an AT command.



Figure 4-2 AT command execution procedures

1. Initialize AT command. Complete registration of the AT command attribute table, command complete cb, and app timer.

### **Note**:

GODIX

- AT command attribute table records AT command information such as AT Command ID, AT Command Tag, AT Command Tag Length, and AT Command Handler.
- command complete cb writes the command execution result to the ble to uart buffer. If an error exists in the command response, the error code is regarded as to-be-transmitted data. If there is no error, the command response data is regarded as to-be-transmitted data.
- app timer manages timeouts.
- 2. Check the received AT command. If the command is to be parsed, change the status of the command from Pending Parsing to In Parsing and perform Step 3. Otherwise, update the error code and perform Step 7.
- Check whether the input AT command is complete. A complete AT command starts with AT: and ends with \r\
   n. If the command is complete, perform Step 4. Otherwise, update the error code, and perform Step 7.
- 4. Get parameters and ID of the AT command.
- 5. Check whether the AT command ID is valid. If the command ID is valid, change the status of the command to Pending Execution. Otherwise, update the error code, and perform Step 7.
- 6. Execute the AT command. If the AT Command Handler is not null, call the handler to perform related Bluetooth LE operation. Otherwise, update the error code, and perform Step 7.
- 7. Call command complete cb to write the command execution result to the ble to uart buffer.

### 4.2 Major Code

This section introduces the major code of the ble\_app\_uart\_at example.

### 4.2.1 Event Handler Function

**Path:**user\_app\user\_app.c

Name: gus\_service\_process\_event();

When a data receiving event occurs, the event handler function checks whether the received data is an AT command. If yes, call at\_cmd\_parse. If no, write the data to ble to uart buffer.

```
void at_cmd_schedule(void)
{
    uint8_t ble_rx_data[AT_CMD_BUFFER_SIZE_MAX];
    switch (p_evt->evt_type)
    {
        ...
        case GUS_EVT_RX_DATA_RECEIVED:
            if (0 == memcmp(p_evt->p_data, "AT:", 3))
            {
             memcpy(ble_rx_data, p_evt->p_data, p_evt->length);
        }
    }
}
```



```
if ((0x0d ! = p_evt->p_data[p_evt->length - 2]) ||\
        (0x0a ! = p_evt->p_data[p_evt->length - 1]))
        {
            ble_rx_data[p_evt->length] = 0x0d;
            ble_rx_data[p_evt->length + 1] = 0x0a;
        }
        at_cmd_parse(AT_CMD_SRC_BLE, ble_rx_data, p_evt->length + 2);
        }
        else
        {
            ble_to_uart_buff_data_push(p_evt->p_data, p_evt->length);
        }
        break;
....
}
```

### 4.2.2 Checking and Updating Environment Variables of AT Command

Path: gr\_libraries\at\_cmd.c under the project directory

Name: at\_cmd\_parse();

}

If the received data is an AT command, check and update the environment variables of the command. If the command passes all checks, set the environment variable state of the command to Pending Execution (AT\_CMD\_IN\_WAITE\_EXECUTE). The code snippet is as follows:

```
void at cmd parse(at cmd src t cmd src, const uint8 t *p data, uint16 t length)
{
    AT CMD RSP DEF(cmd rsp);
    s_at_cmd_env.cmd_src = cmd_src;
    // Check parse cmd is allowed or not
    if (AT CMD IN READY PARSE ! = s at cmd env.cmd state)
    {
        cmd rsp.error code = AT CMD ERR PARSE NOT ALLOWED;
        at_cmd_execute_cplt(&cmd_rsp);
        return;
    }
    else
    {
        s at cmd env.cmd state = AT CMD IN PARSING;
    // Check cmd input is integrity or not
    if (!at cmd integrity check(p data, length, &s parse rlt))
    {
        cmd rsp.error code = AT CMD ERR INVALID INPUT;
        at_cmd_execute_cplt(&cmd_rsp);
        return;
    }
    // Get cmd parameters
    at cmd args get(&s parse rlt);
    // Get cmd Id
    at_cmd_id_get(&s_parse_rlt);
```

# GODIX

```
// Check cmd id is valid or not
if (AT_CMD_INVALID == s_parse_rlt.cmd_id)
{
    cmd_rsp.error_code = AT_CMD_ERR_UNSUPPORTED_CMD;
    at_cmd_execute_cplt(&cmd_rsp);
    return;
}
s_at_cmd_env.cmd_state = AT_CMD_IN_WAITE_EXECUTE;
}
```

### 4.2.3 Performing Bluetooth LE Operations Specific to AT Command

Path: gr\_libraries\at\_cmd.c under the project directory

```
Name: at_cmd_schedule();
```

Read the environment variables of the AT command. If the command is in Pending Execution state with the Command Handler being not null, perform Bluetooth LE operations such as advertising, scanning, and connection based on the attribute table of the AT command. If the AT command is used to modify device name, call the command handler function, uart\_at\_gap\_name\_set, to modify the device name.

```
void at cmd schedule(void)
    if (AT CMD IN WAITE EXECUTE == s at cmd env.cmd state)
    {
        s at cmd env.cmd state = AT CMD IN EXECUTING;
        if (s at cmd env.p cmd attr[s parse rlt.cmd idx].cmd handler)
        {
            if (s_at_cmd_env.cmd time cb)
            {
                s_at_cmd_env.cmd_time_cb();
            }
            s at cmd env.p cmd attr[s parse rlt.cmd idx].cmd handler(&s parse rlt);
        }
        else
        {
            AT CMD RSP DEF(cmd rsp);
            cmd rsp.error code = AT CMD ERR NO CMD HANDLER;
            at_cmd_execute_cplt(&cmd_rsp);
        }
    }
}
```

Path: user\_app\at\_cmd\_handler.c under the project directory

Name: uart\_at\_gap\_name\_set();

The code snippet is as follows:

```
void uart_at_gap_name_set(at_cmd_parse_t *p_cmd_param)
{
    AT_CMD_RSP_DEF(cmd_rsp);
    sdk_err_t error_code;
    uint32_t index;
    if (2 ! = p cmd param->arg count)
```

# GODiX

```
cmd rsp.error code = AT CMD ERR INVALID PARAM;
    }
    else
    {
        if (at_cmd_decimal_num_check(&p_cmd_param->p_buff[p_cmd_param->arg_idx[0]],
                                            p_cmd_param->arg_length[0],&index))
        {
            error code = ble gap device name set((gap dev name write perm t)index,
                   &p cmd param->p buff[p cmd param->arg idx[1]],p cmd param-
>arg length[1]);
            cmd rsp.error code = at cmd ble err convert(error code);
        }
        else
        {
            cmd rsp.error code = AT CMD ERR INVALID PARAM;
        }
    }
    if (AT CMD ERR NO ERROR == cmd rsp.error code)
        cmd rsp.length = at cmd printf bush(cmd rsp.data, "OK");
    at cmd execute cplt(&cmd rsp);
```

### 4.2.4 Writing AT Command Execution Result to ble to uart Buffer

```
Path: gr_libraries\at_cmd.c under the project directory
```

Name: at\_cmd\_execute\_cplt();

After the AT Command Handler is executed successfully, update the to-be-transmitted data based on the return value of the AT command. Call cmd\_cplt\_cb to write the execution result of the command to the ble to uart buffer. The code snippet is as follows:

```
void at cmd execute cplt(at cmd rsp t *p cmd rsp)
{
    uint8 t length = 0;
    if (AT CMD ERR NO ERROR != p cmd rsp->error code)
    {
        switch(p_cmd_rsp->error code)
        {
            case AT CMD ERR UNSUPPORTED CMD:
                length = at cmd printf bush(at cmd rsp buff, "ERR: Unsupported AT CMD.");
                break;
             . . .
         }
    }
    else
    {
        memcpy(at cmd rsp buff, p cmd rsp->data, p cmd rsp->length);
        length = p cmd rsp->length;
    at cmd rsp buff[length] = 0x0d;
```

# GODiX

```
at_cmd_rsp_buff[length + 1] = 0x0a;

if (s_at_cmd_env.cmd_cplt_cb)

{

    if (AT_CMD_SRC_UART == s_at_cmd_env.cmd_src)

    {

        s_at_cmd_env.cmd_cplt_cb(AT_CMD_RSP_DEST_UART, at_cmd_rsp_buff, length + 2);

    }

    else if (AT_CMD_SRC_BLE == s_at_cmd_env.cmd_src)

    {

        s_at_cmd_env.cmd_cplt_cb(AT_CMD_RSP_DEST_BLE, at_cmd_rsp_buff, length + 2);

    }

    s_at_cmd_env.cmd_state = AT_CMD_IN_READY_PARSE;

    ....

}
```

```
Path: user_app\at_cmd_handler.c under the project directory
```

#### Name: user\_at\_cmd\_callback();

### 4.2.5 Reading Ring Buffers and Transmitting Data

Path: user\_app\transport\_scheduler.c under the project directory

Name: transport\_schedule();

When Notify and Flow Control characteristics on the device are enabled, if data exists in the ring buffers, read the data; then transmit the read data.

```
void transport_schedule(void)
{
    uint16_t items_avail = 0;
    uint16_t read_len = 0;
    // read data from s_uart_to_ble_buffer, then notify or write to peer.
    if (transport_flag_cfm(GUS_TX_NTF_ENABLE) && transport_flag_cfm(BLE_TX_CPLT) &&
        transport_flag_cfm(BLE_TX_FLOW_ON))
    {
        items_avail = ring_buffer_items_count_get(&s_uart_to_ble_buffer);
    }
}
```

#### Application Details

```
GODIX
```

}

```
if (items_avail > 0)
    {
        read_len = ring_buffer_read(&s_uart_to_ble_buffer, s_ble_tx_data,
                                      s_mtu_size - 3);
        transport flag set(BLE TX CPLT, false);
        if (BLE GAP ROLE PERIPHERAL == uart at curr gap role get())
        {
            gus_tx_data_send(0, s_ble_tx_data, read_len);
        }
        else if (BLE_GAP_ROLE_CENTRAL == uart_at_curr_gap_role_get())
        {
            gus_c_tx_data_send(0, s_ble_tx_data, read_len);
        }
    }
}
// read data from s_ble_to_uart_buffer, then send to uart.
items_avail = ring_buffer_items_count_get(&s_ble_to_uart_buffer);
if (items_avail > 0)
{
    read_len = ring_buffer_read(&s_ble_to_uart_buffer, s_uart_tx_data,
                                ONCE_SEND_DATA_SIZE);
    uart_tx_data_send(s_uart_tx_data, read_len);
}
```

## **5 Custom Commands**

This chapter depicts how to customize AT commands when using and verifying the ble\_app\_uart\_at example.

Add necessary elements of custom commands into the AT command attribute table. The elements include AT Command ID, AT Command, AT Command Length, and AT Command Handler. After that, implement the Command Handler(s).

#### Note:

You can find the AT command attribute table in user\_app\at\_cmd\_handler.c under the project directory.

Take the custom AT command for MTU exchange as an example. Follow the steps below to add the command to the attribute table.

- Add the required AT Command ID to the at\_cmd\_id\_t structure in the *at\_cmd.h* file (available in SDK\_Folder\ components\libraries\at\_cmd).
- Update the AT command attribute table in code, and add necessary command elements to s\_at\_cmd\_attr\_table. The updated AT command attribute table is displayed below:

```
static at cmd attr t s at cmd attr table[] =
         {AT CMD INVALID,
                                                                  "",
                                                                                                   0, NULL},
                                                                 "TEST",
         {AT CMD TEST,
                                                                                                uart at test},
         {AT_CMD_TEST, "TEST", 4, uart_at_test},
{AT_CMD_VERSION_GET, "VERSION?", 8, uart_at_version_get},
                                                               "RESET",
         {AT CMD RESET,
                                                                                                  5, uart at app reset},
                                                                                          5, uart_at_baud_set},
5, uart_at_bd_addr_get},
", 9, uart_at_gap_role_get},

         {AT_CMD_BAUD_SET,
{AT_CMD_ADDR_GET,
                                                              "BAUD=",
                                                             "ADDR?",
        {AT_CMD_ADDR_GET, "ADDR?", 5, uart_at_bu_addr_get,
{AT_CMD_GAP_ROLE_GET, "GAP_ROLE?", 9, uart_at_gap_role_get},
{AT_CMD_GAP_ROLE_SET, "GAP_ROLE=", 9, uart_at_gap_role_set},
{AT_CMD_GAP_NAME_GET, "GAP_NAME?", 9, uart_at_gap_name_get},
{AT_CMD_GAP_NAME_SET, "GAP_NAME=", 9, uart_at_gap_name_set},
{AT_CMD_ADV_PARAM_SET, "ADV_PARAM=", 10, uart_at_gap_name_set},
{AT_CMD_ADV_START, "ADV_START", 9, uart_at_adv_start},
{AT_CMD_ADV_STOP, "ADV_STOP", 8, uart_at_adv_stop},
         {AT_CMD_SCAN_PARAM_SET, "SCAN_PARAM=", 11, uart_at_scan_param_set},
{AT_CMD_SCAN_START, "SCAN_START", 10, uart_at_scan_start},
{AT_CMD_SCAN_STOP, "SCAN_STOP", 9, uart_at_scan_stop},
        {AT_CMD_SCAN_STOP, "SCAN_STOP", 9, uart_at_scan_stop},
{AT_CMD_CONN_PARAM_SET, "CONN_PARAM=", 11, uart_at_conn_param_set},
{AT_CMD_CONN_INIT, "CONN_INIT=", 10, uart_at_conn_init},
{AT_CMD_CONN_CANCEL, "CONN_CANCEL", 11, uart_at_conn_cancle},
{AT_CMD_DISCONN, "DISCONN", 7, uart_at_disconnect},
                                                                                                    9, uart_at_scan_stop},
         {AT CMD MTU EXCHANGE, "MTU_EXC", 7, uart_at_mtu_exchange},
};
```

#### 🛄 Note:

Added code is in bold.

#### 3. Implement AT Command Handler.

```
void uart_at_mtu_exchange(at_cmd_parse_t *p_cmd_param)
{
    AT_CMD_RSP_DEF(cmd_rsp);
    sdk_err_t error_code;
```

# GODIX

}

```
error_code = ble_gattc_mtu_exchange(0);
cmd_rsp.error_code = at_cmd_ble_err_convert(error_code);
if (AT_CMD_ERR_NO_ERROR ! = cmd_rsp.error_code)
{
    at_cmd_execute_cplt(&cmd_rsp);
}
```

# 6 FAQ

# 6.1 Why Do I Fail to Set GAP Roles Through AT Commands?

Description

Users cannot set GAP roles by using AT commands.

Analysis

The device is not in standby state, resulting in failure to set GAP roles through AT commands.

Solution

Ensure the device is in standby state when using AT commands to set GAP roles.

# 6.2 Why Do I Fail to Set Device Information?

Description

Users cannot set device information by using AT commands.

Analysis

When setting device information such as modifying GAP roles and names, a space exists after "=" in an AT command.

Solution

Make sure there is no space after "=" in an AT command.

# 6.3 Why Does "Invalid Input" Prompt Occur When Users Type an AT Command into GRUart?

Description

GRUart prompts that the input AT command is invalid.

Analysis

An AT command shall end with r n. When users type an AT command on GRUart, **NewLine** in the **Single** tab under **Send data** is unchecked.

Solution

Remember to check NewLine in the Single tab under Send data on GRUart.

# 7 Appendix

# 7.1 AT Command Table

The table below shows the AT commands involved in the ble\_app\_uart\_at example.

Table 7-1 Supported AT commands for ble\_app\_uart\_at

| Command Type   | AT Command                                       | Description  | Return Value  | Example          |
|----------------|--|--|---|------------------|
| Test           | AT:TEST  | Tests whether AT command operates properly.  | ОК  | AT:TEST          |
| Version        | AT:VERSION?                                      | Gets the device version number.  | Version number  | AT:VERSION?      |
| System reset   | AT:RESET   | Resets system.   | -   | AT:RESET         |
| Baud rate      | AT:BAUD=<br><new_value></new_value>              | Configures baud rate.<br>NEW_VALUE: baud rate; range: [0,<br>2000000]  | Successful: OK<br>Failed: ERR: Invalid<br>parameters.                                       | AT:BAUD=4900     |
| Device address | AT:ADDR?   | Gets device address.   | Successful: Device address<br>Failed: No device<br>information is returned.                 | AT:ADDR?         |
| GAP role       | AT:GAP_ROLE?                                     | Gets role information of the device.   | Device roles including<br>NONE, OBSERVER,<br>BROADCASTER,<br>CENTRAL,<br>PERIPHERAL,<br>ALL | AT:GAP_ROLE?     |
|                | AT:GAP_ROLE=<br><new_role></new_role>            | Sets device role.<br>NEW_ROLE: device role. Options<br>include N, n, O, o, B, b, C, c, P, p, A,<br>and a.  | Successful: OK<br>Failed: ERR: Command<br>request is not allowed.                           | AT:GAP_ROLE=O    |
|                | AT:GAP_NAME?                                     | Gets device name.  | Successful: Device name<br>Failed: Specific error code                                      | AT:GAP_NAME?     |
| GAP name       | AT:GAP_NAME= <index,<br>NEW_NAME&gt;</index,<br> | <ul> <li>Sets the device name.</li> <li>INDEX: Write permission of the device name; options: 0, 1, 2, 3, and 4</li> <li>0: Write not allowed</li> <li>1: Link neither encrypted nor authenticated</li> </ul> | Successful: OK<br>Failed: Specific error code   | AT:GAP_NAME=1,Go |



| Command Type | AT Command   | Description  | Return Value                                  | Example                    |
|--------------|--|--|---|----------------------------|
|              |  | <ul> <li>2: Link encrypted but not<br/>authenticated</li> <li>3: Link encrypted and<br/>authenticated (MITM)</li> <li>4: Link encrypted and<br/>authenticated (secure<br/>connections)</li> <li>NEW_NAME: Custom advertising<br/>name</li> </ul>   |   |                            |
| Advertise    | AT:ADV_PARAM=<br><adv_interval,<br>ADV_DURATION&gt;</adv_interval,<br>     | <ul> <li>Sets advertising parameters.</li> <li>ADV_INTERVAL: Advertising<br/>interval; unit: 0.625 ms; range: &gt;<br/>32.</li> <li>ADV_DURATION: Advertising<br/>duration; unit: 10 ms;</li> <li>When ADV_DURATION = 0, the<br/>device will continue advertising<br/>until the host disables it. For Limited</li> <li>Discoverable Mode, the parameter<br/>value ranges from 1 to 18000; for<br/>directed advertising with high duty<br/>cycle, the parameter value ranges</li> <li>from 1 to 128.</li> </ul> | Successful: OK<br>Failed: Specific error code | AT:ADV_PARAM=80,0          |
|              | AT:ADV_START   | Starts advertising.  | Successful: OK<br>Failed: Specific error code | AT:ADV_START               |
|              | AT:ADV_STOP  | Stops advertising.   | Successful: OK<br>Failed: Specific error code | AT:ADV_STOP                |
| Scan         | AT:SCAN_PARAM=<br><scan_interval,<br>SCAN_DURATION&gt;</scan_interval,<br> | <ul> <li>Sets scanning parameters.</li> <li>SCAN_INTERVAL: Scanning<br/>interval; unit: 0.625 ms; range: 4<br/>to 16384</li> <li>SCAN_DURATION: Scanning<br/>duration; unit: 0.625 ms; range: 1<br/>to 65535</li> </ul>  | Successful: OK<br>Failed: Specific error code | AT:SCAN_PARAM=<br>176,1000 |
|              | AT:SCAN_START  | Starts scanning.   | Successful: OK<br>Failed: Specific error code | AT:SCAN_START              |
|              | AT:SCAN_STOP   | Stops scanning.  | Successful: OK                                | AT:SCAN_STOP               |



| Command Type | AT Command  | Description   | Return Value  | Example                     |
|--------------|---|---|---|-----------------------------|
|              |   |   | Failed: Specific error code                             |                             |
| Connect      | AT:CONN_PARAM=<br><conn_interval,<br>CONN_LATENCY,<br/>CONN_SUP_TIMEOUT&gt;</conn_interval,<br> | <ul> <li>Sets connection parameters.</li> <li>CONN_INTERVAL: Connection<br/>interval; range: 6 to 3200</li> <li>CONN_LATENCY: Number<br/>of connection events that<br/>can be ignored; range: &lt;<br/>(CONN_SUP_TIMEOUT/<br/>CONN_INTERVAL) – 1</li> <li>CONN_SUP_TIMEOUT:<br/>Supervision timeout; range: 10 to<br/>3200</li> </ul> | Successful: OK<br>Failed: Specific error code           | AT:CONN_PARAM=<br>12,5,3200 |
|              | AT:CONN_INIT=   | Initiates a connection.   | Successful: CONNECTED<br>Failed: Specific error code    | AT:CONN_INIT=               |
|              | AT:CONN_CANCEL  | Terminates a connection.  | Successful: OK<br>Failed: Specific error code           | AT:CONN_CANCEL              |
| Disconnect   | AT:DISCONN  | Disconnects a connection.   | Successful: DISCONNECTED<br>Failed: Specific error code | AT:DISCONN                  |
| MTU exchange | AT:MTU_EXC  | Exchanges an MTU.   | Successful: MTU<br>Failed: Specific error code          | AT:MTU_EXC                  |

# 7.2 Error Code

When a failure occurs during executing AT commands, error code will be returned. The table below lists error code involved when using and verifying the ble\_app\_uart\_at example.

| Description   |
|---|
| The input information is invalid.   |
| The input AT command is not supported.  |
| The AT command cannot be parsed.  |
| The command request is not allowed; for instance, if the device is not in standby state when users set GAP roles through AT commands, AT_CMD_ERR_CMD_REQ_ALLOWED is returned. |
| The AT Command Handler is null.   |
| The input AT command parameter is invalid.  |
| A timeout occurs for HAL operations.  |
| A timeout occurs when the AT command is executed.   |
|   |

Table 7-2 Error code definitions

# GODIX

| Name                   | Description  |
|------------------------|--------------|
| AT_CMD_ERR_OTHER_ERROR | Other errors |